

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
 - TEXT CUT OFF AT TOP, BOTTOM OR SIDES
 - FADED TEXT
 - ILLEGIBLE TEXT
 - SKEWED/SLANTED IMAGES
 - COLORED PHOTOS
 - BLACK OR VERY BLACK AND WHITE DARK PHOTOS
 - GRAY SCALE DOCUMENTS
-

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平11-3432

(43) 公開日 平成11年(1999) 1月6日

(51) Int.Cl.⁶

識別記号

F I

G 0 6 T 11/00

G 0 6 F 15/72

3 5 0

A 6 3 F 9/22

A 6 3 F 9/22

C

G 0 6 T 15/70

G 0 6 F 15/62

3 4 0 K

15/00

15/72

4 5 0 A

審査請求 未請求 請求項の数13 O L (全 9 頁)

(21) 出願番号 特願平9-155540

(22) 出願日 平成9年(1997) 6月12日

(71) 出願人 000132471

株式会社セガ・エンタープライゼス

東京都大田区羽田1丁目2番12号

(72) 発明者 高野 豪

東京都大田区羽田1丁目2番12号 株式会

社セガ・エンタープライゼス内

(72) 発明者 松本 功

東京都大田区羽田1丁目2番12号 株式会

社セガ・エンタープライゼス内

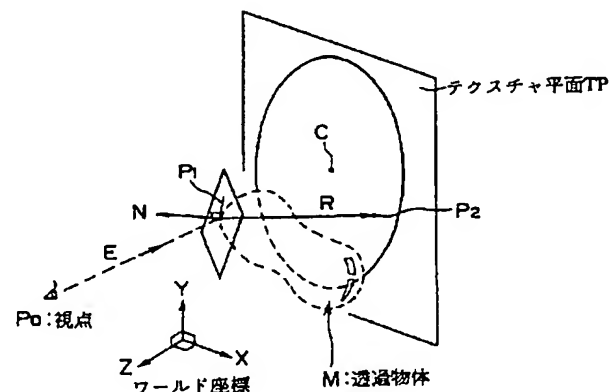
(74) 代理人 弁理士 稲葉 良幸 (外2名)

(54) 【発明の名称】 画像処理装置、ゲーム装置、その方法および記録媒体

(57) 【要約】

【課題】 透明な物体を質感豊かにかつ高速に表示できる画像処理装置およびその方法を提供する。

【解決手段】 空間を透視する視点から透過物体Mへの視線を示す視線ベクトルEを生成し、当該視線ベクトルEと透過物体Mの表面との交点である第1交点P1における法線ベクトルNを生成し、視線ベクトルEと法線ベクトルNとに基づいて、視線ベクトルEに相当する光軸が透過物体Mへ入射して屈折する方向を示す屈折ベクトルRを生成するベクトル演算手段と、特定のテクスチャを設定したテクスチャ平面TPとベクトル演算手段により生成された屈折ベクトルRとの交点である第2交点P2を特定し、特定した第2交点P2に設定されているテクスチャデータに基づいて、透過物体Mの第1交点P1付近におけるテクスチャを決定するレンダリング手段とを備える。リアルタイムに透明体の表示ができる。



【特許請求の範囲】

【請求項 1】 仮想空間において背景が透過して見える透過物体を表示する画像処理装置であって、前記仮想空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、当該視線ベクトルと前記透過物体の表面との交点である第 1 交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第 1 交点における屈折方向を示す屈折ベクトルを計算するベクトル演算手段と、

特定の空間位置に予め設けたテクスチャ平面と前記ベクトル演算手段により計算された前記屈折ベクトルとの交点である第 2 交点を特定し、特定した当該第 2 交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第 1 交点付近におけるテクスチャデータを決定するレンダリング手段と、を備えた画像処理装置。

【請求項 2】 前記ベクトル演算手段は、前記透過物体の表面を形成する複数の微小多角形の各頂点を前記第 1 交点とし、当該第 1 交点を端点とする複数の辺同士の外積に基づいて当該第 1 交点における法線ベクトルを計算し、当該第 1 交点についての前記屈折ベクトルを計算すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 3】 前記ベクトル演算手段は、前記透過物体の表面を形成する複数の微小多角形の各頂点を前記第 1 交点とし、当該第 1 交点を構成する前記微小多角形に予め設定された法線ベクトルを平均して当該第 1 交点における法線ベクトルを計算し、当該第 1 交点についての前記屈折ベクトルを計算すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 4】 前記ベクトル演算手段は、前記透過物体の表面を形成する複数の微小多角形の各頂点を前記第 1 交点とし、当該頂点に予め設定された法線ベクトルに基づいて、当該第 1 交点についての前記屈折ベクトルを計算すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 5】 前記レンダリング手段は、前記微小多角形の各頂点について掲載された前記屈折ベクトルの各々について前記第 2 交点を特定し、特定された各前記第 2 交点により囲まれる領域の前記テクスチャ平面に設定されたテクスチャデータを、当該微小多角形にマッピングするテクスチャデータとする請求項 2 乃至請求項 4 のいずれか一項に記載の画像処理装置。

【請求項 6】 前記ベクトル演算手段は、前記視線ベクトルを E 、前記法線ベクトルを N 、前記透過物体の外側の屈折率を n_1 、前記透過物体の内側の屈折率を n_2 、前記視線ベクトルと前記法線ベクトルとのなす角度 θ_1 、および、前記法線ベクトルと前記屈折ベクトルのなす角度を θ_2 とした場合に、前記屈折ベクトル R を、 $R = (n_1 / n_2) \cdot (E - N(n \cdot \cos \theta_2 - \cos \theta_1))$
 $n = n_2 / n_1$

$$\cos \theta_1 = -E \cdot N$$

$$\cos \theta_2 = 1 - (1 - \cos \theta_1^2) / n^2$$

という演算により求める請求項 1 に記載の画像処理装置。

【請求項 7】 前記レンダリング手段は、前記屈折ベクトルのうち前記テクスチャ平面に平行な成分を参照して、当該テクスチャ平面上の前記第 2 交点の座標を特定することを特徴とする請求項 1 に記載の画像処理装置。

【請求項 8】 前記テクスチャ平面に設定するテクスチャデータは、前記所定の半球面上に設定された画像を予め所定の平面に投射した画像に基づいて生成すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 9】 前記テクスチャ平面に設定するテクスチャデータは、実空間における周囲の景色を映し込ませた球状体を一点から撮影した画像に基づいて生成すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 10】 前記レンダリング手段は、前記視線ベクトルの方向に対応させて前記テクスチャ平面に設定するテクスチャデータを変更すること、を特徴とする請求項 1 に記載の画像処理装置。

【請求項 11】 請求項 1 乃至請求項 10 に記載の画像処理装置を備え、当該画像処理装置により生成された仮想物体を、ゲームの表示対象とするゲーム装置。

【請求項 12】 仮想的な空間において光が透過する透過物体を表現する画像処理方法であって、前記空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、

前記視線ベクトルと前記透過物体の表面との交点である第 1 交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第 1 交点における屈折方向を示す屈折ベクトルを生成する屈折ベクトル生成工程と、

特定の空間位置に設けたテクスチャ平面と前記屈折ベクトル生成工程で生成された前記屈折ベクトルとの交点である第 2 交点を特定する交点特定工程と、

前記交点特定工程で特定した当該第 2 交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第 1 交点付近におけるテクスチャデータを決定するレンダリング工程と、を備えた画像処理方法。

【請求項 13】 仮想的な空間において光が透過する透過物体を表現する画像処理を実行させるプログラムであって、

前記空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、前記視線ベクトルと前記透過物体の表面との交点である第 1 交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第 1 交点における屈折方向を示す屈折ベクトルを生成する屈折ベクトル生成工程と、

特定の空間位置に設けたテクスチャ平面と前記屈折ベクトル生成工程で生成された前記屈折ベクトルとの交点で

ある第2交点を特定する交点特定工程と、前記交点特定工程で特定した当該第2交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第1交点付近におけるテクスチャデータを決定するレンダリング工程と、をコンピュータに実行させるプログラムが記録された機械読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、ビデオゲーム装置等に適用される画像処理技術に係り、特に、光が透過する透明な物体を質感豊かに表示するための画像処理技術に関する。

【0002】

【従来の技術】画像処理技術の発達に伴い、実際に存在しない物体を立体的に表現できるようになった。特に、ビデオゲーム装置の技術分野では、仮想的な三次元空間（以下「仮想空間」という）の画像を高速に表示することが行われている。仮想空間に物体を表示するためには、通常、照明や環境からの光の反射をシミュレーションする。ところで、このゲームの内容によっては、ガラスや水のような光を透過する液体や固体、流動体（以下「透過物体」という）を表示する必要が生ずる。従来の画像処理技術において、透明物体を表示するためには、幾つかの方法が考えられていた。

【0003】最も簡単に透過物体を表示方法は、光が透過物体に入射することにより生ずる屈折を無視することである。この場合、画像の表示処理は高速に行えるであろうが、屈折を無視したのでは、実際に透明な物体を透過した場合に観察される背景の画像の歪みが表現できない。

【0004】このため、現実的な透明感を十分に表現するための方法としては、レイ・トレーシング・アルゴリズム等のレンダリング（rendering）手法により光の屈折を考慮することが、行われている。空気中から水中に光線が入る場合のように、光がある媒質から他の媒質へ入射するときには屈折が生ずる。この光線が曲げられる量は、一般にスネル（Snell）の法則により関係づけられる。レイ・トレーシング・アルゴリズムでは、画像を表示するための観察者の視点からの視線を逆に辿り、光を反射する物体では光を反射させ、光を透過する透過物体では、前記スネルの法則により屈折方向を求め、視点から観察できる画像を定める。

【0005】また、ゲーム制作の現場においては、例えば格闘ゲームにおいて、特殊なキャラクタ（ゲームの表示対象となる人物、ロボット等）を透明感をもって表示し、ゲームの特殊性を一層際立たせるために、上記レンダリング手法を付加しようとする試みがなされていた。

【0006】

【発明が解決しようとする課題】しかしながら、上記したレイ・トレーシングによるレンダリングでは、陰影、

屈折や反射をすべて正確に再現できる利点があるが、演算量が膨大なものとなるため、ビデオゲーム装置のような高速処理を主要な技術的課題の一つとする装置には、不適当な方法であるといえる。

【0007】そこで、本発明は、質感豊かにかつ高速に透明な物体を表示できる画像処理装置、その方法および記録媒体を提供することを第1の課題とする。

【0008】また、これらの画像処理技術を用い、特殊な表示形態のキャラクタをより効果的にリアルタイム表現するゲーム装置を提供することを第2の課題とする。

【0009】

【課題を解決するための手段】請求項1に記載の発明は、仮想空間において背景が透過して見える透過物体を表示する画像処理装置である。すなわち、前記仮想空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、当該視線ベクトルと前記透過物体の表面との交点である第1交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第1交点における屈折方向を示す屈折ベクトルを計算するベクトル演算手段を備える。さらに、特定の空間位置に予め設けたテクスチャ平面と前記ベクトル演算手段により計算された前記屈折ベクトルとの交点である第2交点を特定し、特定した当該第2交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第1交点付近におけるテクスチャデータを決定するレンダリング手段を備える。

【0010】前記ベクトル演算手段は、請求項2乃至請求項4に記載したように、前記透過物体の表面を形成する複数の微小多角形の各頂点を前記第1交点とするのが好ましい。

【0011】請求項2の発明では、当該第1交点を端点とする複数の辺同士の外積に基づいて当該第1交点における法線ベクトルを計算する。請求項3に記載の発明では、当該第1交点を構成する前記微小多角形に予め設定された法線ベクトルを平均して当該第1交点における法線ベクトルを計算する。請求項4に記載の発明では、当該頂点に予め設定された法線ベクトルに基づいて、当該第1交点についての前記屈折ベクトルを計算する。

【0012】前記レンダリング手段は、請求項5に記載したように、前記微小多角形の各頂点について掲載された前記屈折ベクトルの各々について前記第2交点を特定し、特定された各前記第2交点により囲まれる領域の前記テクスチャ平面に設定されたテクスチャデータを、当該微小多角形にマッピングするテクスチャデータとするのが好ましい。

【0013】前記ベクトルの演算は、請求項6に記載したように、前記視線ベクトルをE、前記法線ベクトルをN、前記透過物体の外側の屈折率を n_1 、前記透過物体の内側の屈折率を n_2 、前記視線ベクトルと前記法線ベクトルとのなす角度 θ_1 、および、前記法線ベクトルと

前記屈折ベクトルのなす角度を θ_2 とした場合に、前記屈折ベクトル R を、

$$R = (n_1 / n_2) \cdot (E - N(n \cdot \cos \theta_2 - \cos \theta_1))$$

$$n = n_2 / n_1$$

$$\cos \theta_1 = -E \cdot N$$

$$\cos \theta_2 = 1 - (1 - \cos \theta_1^2) / n^2$$

という演算により求めることができる。

【0014】また、請求項7に記載の発明のように、前記屈折ベクトルのうち前記テクスチャ平面に平行な成分を参照して、当該テクスチャ平面上の前記第2交点の座標を特定するのは好ましい。例えば、テクスチャ平面に平行な方向を u 方向および v 方向とする。前記屈折ベクトルを当該テクスチャ平面へ投射した場合の u 方向および v 方向のベクトル成分をそれぞれ R_u および R_v とすると、前記第2交点の座標 (u, v) は、

$$u = (R_u + 1) / 2$$

$$v = (R_v + 1) / 2$$

というような演算により求められる。この場合、テクスチャデータは、透過物体の中心を通るような屈折がまったくない場合の視線ベクトルが、このテクスチャ平面の中心点を通過するように、設定しておくことになる。

【0015】このテクスチャ平面に設定するテクスチャデータは、請求項8に記載したように、所定の半球面上に設定された画像を予め所定の平面に投射して生成するのが好ましい。また、請求項9に記載したように、実空間における周囲の景色を映し込ませた球状体を一点から撮影して生成するのも好ましい。

【0016】さらに、前記レンダリング手段は、請求項10に記載したように、前記視線ベクトルの方向に対応させて前記テクスチャ平面に設定するテクスチャデータを変更するのも好ましい。

【0017】請求項11に記載の発明は、第2の目的を達成するためのものである。すなわち、当該ゲーム装置は、請求項1乃至請求項10に記載の画像処理装置を備え、当該画像処理装置により生成された仮想物体を、ゲームの表示対象とする。

【0018】請求項12に記載の発明は、仮想的な空間において光が透過する透過物体を表現する画像処理方法であって、前記空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、前記視線ベクトルと前記透過物体の表面との交点である第1交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第1交点における屈折方向を示す屈折ベクトルを生成する屈折ベクトル生成工程と、特定の空間位置に設けたテクスチャ平面と前記屈折ベクトル生成工程で生成された前記屈折ベクトルとの交点である第2交点を特定する交点特定工程と、前記交点特定工程で特定した当該第2交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第1交点付近におけるテクスチャデ

ータを決定するレンダリング工程と、を備える。

【0019】請求項13に記載の発明は、仮想的な空間において光が透過する透過物体を表現する画像処理を実行させるプログラムであって、前記空間を透視する視点から前記透過物体への視線を示す視線ベクトルと、前記視線ベクトルと前記透過物体の表面との交点である第1交点における法線ベクトルと、に基づいて、前記視線ベクトルに平行な光の前記第1交点における屈折方向を示す屈折ベクトルを生成する屈折ベクトル生成工程と、特定の空間位置に設けたテクスチャ平面と前記屈折ベクトル生成工程で生成された前記屈折ベクトルとの交点である第2交点を特定する交点特定工程と、前記交点特定工程で特定した当該第2交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第1交点付近におけるテクスチャデータを決定するレンダリング工程と、をコンピュータに実行させるプログラムが記録された機械読み取り可能な記録媒体である。

【0020】なお、記録媒体としては、フロッピーディスク等の磁気ディスク、CD、CD-ROM、CD-R、書き換え可能光ディスク等の光ディスク、あるいは揮発性若しくは不揮発性のRAM、ROM、ICカード等の半導体メモリ等、機械読み取り可能なあらゆる記憶媒体が相当する。さらに、インターネット等の通信回線を介して遠隔地のコンピュータ装置等からプログラムデータが転送される場合も、ここにいう記憶媒体に含まれる。

【0021】

【発明の実施の形態】以下、本発明の好適な実施の形態を図面を参照して説明する。本実施の形態は、本発明の画像処理装置を適用したゲーム処理装置に関する。

【0022】(I) 構成の説明

図1に、本実施の形態におけるゲーム処理装置のブロック図を示す。同図に示すように本ゲーム処理装置100は、画像処理ブロック1、描画ブロック2、モニタ装置3およびメモリ4をバスライン5で相互に接続して構成される。本発明の画像処理は、画像処理ブロック1において処理される。

【0023】画像処理ブロック1は、CPU101、ROM102、インターフェース回路103、入力装置104および補助ブロック105により構成される。CPU101は、補助ブロック105に装着されたCD-ROM110から読み込まれたプログラムデータに基づいて、本発明の画像処理を行う。ROM102は、電源投入後のIPLプログラム等を格納する。インターフェース回路103は、操作者の操作により入力装置104から入力された操作信号を、CPU101に供給する。補助ブロック105は、装着されたCD-ROM110からプログラムデータおよび画像データを読み込み、メモリ4に転送する。なお、プログラムデータを供給する記憶媒体としては、CD-ROM110に限らず、他の記

憶媒体であってもよい。

【0024】描画ブロック2は、描画プロセッサ201、フレームバッファ202および203により構成される。画像処理プロセッサ201は、画像処理ブロック1のCPU101から表示用画像データが転送されると、これをフレームバッファ202に格納する。表示用画像データは、表示対象である物体を構成する微小多角形であるポリゴンの頂点座標、当該ポリゴンに適用するテクスチャデータを指定する命令データ、当該ポリゴンのワールド座標系における表示座標を特定するデータ、当該ワールド座標を観察する視点の位置を示すデータ等により構成される。描画プロセッサ201は、これらフレームバッファ201に格納されたデータに基づいて、ディスプレイに表示する画像である視野座標系における各ポリゴンの二次元座標をマトリクス演算によって求め、テクスチャデータとしてはメモリ4のテクスチャデータメモリ402を参照し、その座標に基づいて得られるビットマップデータをフレームバッファ203に格納する。そして、フレームバッファ203に格納されたビットマップデータに基づいて、モニタ装置3に映像信号を供給し画像表示を行う。

【0025】メモリ4は、演算に使用されるRAM401およびテクスチャデータを記憶するテクスチャデータメモリ402を備える。RAM401には、本発明の画像処理に必要なベクトル演算のためのデータを格納される。つまり、上記演算前のベクトルを特定するデータを格納し、さらに演算後のベクトルを特定するデータ等が格納される。テクスチャデータメモリ402には、描画ブロック2が表示用ビットマップデータを生成する際に参照されるテクスチャデータが格納される。

【0026】図2に、本実施の形態のゲーム処理装置100を機能面からブロック化した機能ブロック図を示す。上記画像処理ブロック1は、CPU101がCD-ROM110から読み取られたプログラムデータに基づいて動作することにより、当該機能ブロックの協働作用を行う。

【0027】すなわち、画像処理ブロック1は、ベクトル演算手段として作用するベクトル演算ブロック10、およびレンダリング手段として作用するレンダリングブロック20として動作する。ベクトル演算ブロック10は、視線ベクトル生成部11、法線ベクトル生成部12および屈折ベクトル生成部13を備える。レンダリングブロック20は、交点特定部21およびテクスチャ決定部22を備える。

【0028】ベクトル演算ブロック10は、位置情報（ワールド座標系における三次元座標等）からベクトル成分を特定する数値を計算するブロックである。視線ベクトル生成部11は、メモリ4のRAM401に格納された、視点の位置情報と、本発明を適用して光を透過させて表示したい透過物体のポリゴンデータとに基づい

て、視点からこの透過物体上の各ポリゴンへの視線ベクトルを計算し、再びRAM401に格納する。法線ベクトル生成部12は、RAM401に格納されたポリゴンデータに基づいて当該ポリゴンの各頂点における法線ベクトルを計算し、再びRAM401に格納する。屈折ベクトル生成部13は、RAM401に格納された、視線ベクトル生成部11の生成した視線ベクトルと法線ベクトル生成部12の生成した当該ポリゴンの各頂点における法線ベクトルとを参照し、屈折ベクトルを計算し、再びRAM401に格納する。

【0029】レンダリングブロック20は、ベクトル演算ブロック10により計算され、RAM401に格納された屈折ベクトルに基づいて、透過物体に適用するテクスチャデータを決定するブロックである。交点特定部21は、屈折ベクトル生成部13の生成した屈折ベクトルのうち、テクスチャ平面に平行な座標成分を計算する。そしてこれに基づいて、屈折ベクトルとテクスチャ平面との交点（第2交点）の座標をポリゴンの頂点に対応させて計算し、RAM401に格納する。テクスチャ決定部22は、ポリゴンごとに、前記第2交点におけるテクスチャデータの範囲をそれぞれ特定し、当該ポリゴンに適用するテクスチャデータを示す命令データを描画ブロック2に転送する。

【0030】描画ブロック2は、前述したように、視点の位置情報、ポリゴンデータに基づいて、複数のポリゴンにより透過物体等の表示すべき物体を描画する。その際、透過物体について、テクスチャ決定部22の決定したテクスチャデータを、テクスチャデータメモリ402から読み出して、当該ポリゴンの表面にマッピングして表示用のビットマップデータを生成する。

【0031】(II) 動作の説明

次に、本実施の形態の動作を説明する。処理を行う前提として、メモリ4には、仮想画像を観察する視点の位置情報、透過物体等を構成するポリゴンデータ、テクスチャ平面に設定するテクスチャデータ、テクスチャ平面をワールド座標系で定義する情報等が格納されているものとする。

【0032】従来の技術で説明したように、レイ・トレーシング・アルゴリズム等のレンダリング手法により透明物体を表現しようとすれば、その演算量が膨大であるため、実時間処理に不適である。

【0033】ここで、実時間処理を行うために、視点からの光線の追跡を一回に限り行うことも考えられる。すなわち、視線からの光線の透過物体の表面における屈折を計算し、屈折した光線の到達する背景画像の画素を特定するのである。この背景画像の画素を透過物体の表面に表示すれば、透過物体が光を透過し、背景画が映り込んでいるように見える。しかし、透過物体の画素ごとに背景画像の画素を特定するものとしても、ビデオゲーム装置等におけるリアルタイム画像処理としては、まだそ

の演算量は膨大であり、実施の製品に適用することが困難である。本発明は、これを克服するため、以下の処理を行う。

【0034】本画像処理装置において、視点の位置は、仮想空間を絶対位置を特定するための三次元座標（ワールド座標系）における座標値として与えられる。ポリゴンの頂点の位置は、物体ごとに定義されるボディ座標系の座標値として与えられる。なお、法線ベクトルを計算で求めずポリゴンごとに予め定義しておくこともできる。この場合には、このポリゴンの頂点ごとの法線ベクトルを特定する情報がCD-ROM110からRAM401に予め転送されることになる。

【0035】テクスチャ平面上に設定するテクスチャデータは、例えば、図7に示すように、反射率の高い金属球に周囲の景色を映し込み、映し込んだ景色をカメラで撮影し、画像データ化して作る。すなわち、透過物体が球形であるとする、視線が透過物体の表面に入射する角度が浅い程（すなわち、透過物体の周囲になる程）光の屈折率が高くなる。したがって、金属球に移る周囲の景色を、透明物体の表面に張り付けるテクスチャデータとして透明物体を構成するポリゴンに張り付ければ、当該透過物体に背景画像が屈折して映っているかのような印象を観察者に与えることができるのである。なお、このような手法によらず、金属体の表面に映り込む画像を、公知のレイ・トレーシングの手法により生成することによって、テクスチャ平面上に設定するテクスチャデータを制作してもよい。

【0036】図8に、金属球を撮影する等により得られたテクスチャデータの例を示す。座標 u 、 v はテクスチャ平面を定義するための座標軸である。この座標軸は、テクスチャ平面に平行なベクトル成分を有する。同図に示すように、テクスチャデータは、テクスチャ平面の中心点 C で疎、周辺で密になるよう設定された画像データである。金属球を撮影してテクスチャデータを撮影した場合等には、同心円状にデータが得られるので、その外側は、濃度の高い画素データ等で埋める。

【0037】図3に、本形態の動作を説明するフローチャートを示す。画像処理ブロック1は、所定の画像書換えの周期（例えば、ビデオ信号の垂直同期期間）が来ると、モニタ装置3に表示すべき仮想空間の画像を更新する。すなわち、仮想空間内に物体が存在するか否かを判定し（ステップS（以下「S」と表示する）1）、物体が存在する場合（S1；YES）、その物体が、背景画像等の光を透過させて表示する物体であるか否かを判定する（S2）。光を透過させない通常の物体（通常のテクスチャデータが適用される物体）であれば（S2；NO）、通常のテクスチャマッピング処理を行う。光を透過させる透過物体であれば（S2；YES）、本発明に係る以下の処理を続行する。

【0038】まず、この透過物体を構成するポリゴンの

うち一つを特定し（S3）、さらにその特定したポリゴンの複数の頂点のうち一つを特定する（S4）。視線ベクトル生成部11は、メモリ4に格納された視点の座標を参照し、視点からこの頂点へ向かう視線ベクトル E を生成する（S5）。

【0039】この様子を図4および図5に示す。図4は、視線ベクトル等を透過物体の關係の斜視図であり、図5は、ポリゴンの頂点と視線ベクトル、屈折ベクトルを含む平面で透過物体を見た断面図である。同図で、視線ベクトルを E 、法線ベクトルを N （ N' ）、屈折ベクトルを R で表わす。また、視点は $P0$ 、頂点は $P1$ 、テクスチャ平面 TP と屈折ベクトル R との交点を $P2$ で示す。視線ベクトル E は、視点 $P0$ の座標とポリゴンの頂点 $P1$ の座標から求められる。

【0040】次いで、頂点 $P1$ についての法線ベクトル N を求める（S6）。法線ベクトル N は、予め頂点ごとのベクトル値を計算し、計算値をメモリ4に格納しておくてもよい。本実施形態では、図6に示すように、頂点を囲むポリゴンのポリゴンデータから算出するものとする。図6は、頂点 $P1$ を楕円で $S12$ 、 $S13$ 、 $S23$ の3つのポリゴンが取り囲んでいる様子を示す。ポリゴンデータには各ポリゴンの頂点座標が記録されているので、各頂点の座標を参照すれば各頂点を結ぶ辺ベクトル（ $L1$ 、 $L2$ 、 $L3$ ）が求められる。頂点 $P1$ についての法線ベクトル N は、各辺ベクトルの外積を各々計算し、計算した外積の合計を計算することで求められる。

【0041】すなわち、法線ベクトル N は、
$$N = L1 \times L2 + L3 \times L1 + L2 \times L3$$
で求められる。

【0042】法線ベクトル N が求められると、次に屈折ベクトル生成部13は屈折ベクトル R を求める（S7）。屈折ベクトル R を求めるために、予め屈折率を定義しておく。例えば、図5に示すように、透過物体 M の外側の屈折率を $n1$ 、透過物体 M の内側の屈折率を $n2$ とする。また、視線ベクトル E と法線ベクトル N とのなす角度 $\theta1$ および法線ベクトル N' と屈折ベクトル R とのなす角度を $\theta2$ とする。この場合、屈折ベクトル R は、スネルの法則にしたがい、
$$R = (n1/n2) \cdot (E - N(n \cdot \cos \theta2 - \cos \theta1))$$

但し、 $n = n2/n1$ 、 $\cos \theta1 = -E \cdot N$ 、 $\cos \theta2 = 1 - (1 - \cos \theta1^2) / n^2$ という演算式により導き出せる。

【0043】ステップS8において、交点特定部21は、屈折ベクトル R とテクスチャ平面 TP の交点 $P2$ を求める（図4参照）。テクスチャ平面は、計算を簡単にするために、ワールド座標系における z 軸に垂直な面であることが望ましい。すなわち、視線の方向と z 軸方向を一致させた視点座標系における xy 平面に平行なものと仮定する。また、視点 $P0$ と透明物体 M の中心点とテ

クスチャ平面TPの中心点Cとがほぼ一直線上に並ぶようテクスチャ平面の中心点の相対位置を設定するのが好ましい。屈折ベクトルRのうち、テクスチャ平面を定義する座標軸の方向成分(u方向、v方向)と同一方向の成分をRuおよびRvとする。視点P0とテクスチャ平面TPとの距離を適当に選ぶことにより、屈折ベクトルRとテクスチャ平面TPとの交点P2の座標(u, v)は、

$$u = (Ru + 1) / 2$$

$$v = (Rv + 1) / 2$$

という演算により求められる。もちろん、テクスチャ平面が平面方程式で定義され、演算能力に余力がある場合は、屈折ベクトルの示す空間における一次式に基づいて、正確なテクスチャ平面上の交点を求めてもよい。

【0044】なお、仮想平面の背景画像が視線の方向により異なってくるような場合には、テクスチャデータを複数用意しておき、視線の方向に応じて、その背景として透明物体に映し込むテクスチャデータを変更してもよい。

【0045】一つのポリゴンの一つの頂点に対応するテクスチャ平面上の一つの交点が上記S5～S8の工程によって求められる。これを残りの頂点がある限り(S9; YES)、繰り返す。つまり、同一の処理により他の頂点に対応するテクスチャ平面上の交点を求めていく(S5～S8)。一つのポリゴンについてすべての頂点に対応する交点が求められると(S9; NO)、このポリゴンの各頂点に対応して求められたテクスチャ平面上の交点で囲まれた領域を特定する(S10)。これが、当該ポリゴンに対応するテクスチャデータとなる。テクスチャ決定部22は、特定された領域のテクスチャデータを公知のレンダリング手法を用いて、透過物体の対応するポリゴンにマッピングする(S11)。その際、Zバッファ法等の高速処理を用いた隠面処理や各種シェーディング技法を用いてもよい。以上の処理により、一つのポリゴンの処理が完了する。

【0046】以上の処理を、当該透明物体を構成する他のポリゴンについても行う。すなわち、当該透過物体に他の未処理のポリゴンが存在すれば(S12; YES)、その他のポリゴンについて、上記処理(S3～S11)を繰り返し、残りのポリゴンがなくなったら(S12; NO)、他の物体の検索に移行する(S1)。

【0047】図9に、本形態における透過物体のテクスチャマッピングの例を示す。同図に示すように、透過物体を通して背後にあるビル映像を観察する場合、テクスチャデータをこのビルについて作成しておけば、透過物体を通して観察される画像は、光の屈折により得られた現実的な画像となる。

【0048】(III) 効果の説明

以上のように本形態によれば、ポリゴンの頂点ごとに簡単な演算式により、テクスチャデータを特定していくの

で、従来、透過物体を表現するために必要とされた光路計算を省略でき、リアルタイムに現実的な透過物体の表現が行える。

【0049】(IV) その他の形態

本発明は、上記各形態に拘らず種々に変形できる。テクスチャデータの作成の手法は、本形態によらず、実際の屈折率のある物体を近似できるものであれば他の方法を用いてもよい。例えば、実際のガラス球または水晶球のような物体に映りこむ景色の映像を用いてもよい。

【0050】また、屈折ベクトルの演算式、交点を求める演算式も、上記形態によらず、屈折の存在を技術的に特定できる方法によるものであれば、他の方法を用いてもよい。例えば、簡単な演算テーブル等を用いる方法が挙げられる。

【0051】

【発明の効果】本発明によれば、視線ベクトルと法線ベクトルとに基づいて、前記視線ベクトルに平行な光の前記第1交点における屈折方向を示す屈折ベクトルを計算し、特定の空間位置に予め設けたテクスチャ平面と前記屈折ベクトルとの交点である第2交点を特定し、当該第2交点に設定されているテクスチャデータに基づいて、前記透過物体の前記第1交点付近におけるテクスチャデータを決定するので、簡単なベクトル演算により設定すべきテクスチャデータを決められる。したがって、ビデオゲーム装置等のように、高速処理が必要な画像処理装置においても、従来困難とされた透過物体の質感豊かな表現が可能となる。

【0052】特に、テクスチャデータの特定をポリゴンの頂点ごとに行えば、マッピング自体は公知のマッピング技術がそのまま適用できる。

【0053】また、視線ベクトルの方向に対応させてテクスチャデータを更新すれば、視線の方向に応じ、透過物体により現実感のあるテクスチャをマッピングすることができる。

【図面の簡単な説明】

【図1】本発明の実施の形態におけるゲーム装置のブロック図である。

【図2】本発明の実施の形態における画像処理ブロックの機能ブロック図である。

【図3】本発明の実施の形態の動作を説明するフローチャートである。

【図4】視点と透過物体とテクスチャ平面の概念を示す斜視図である。

【図5】視線ベクトルと法線ベクトルと屈折ベクトルとの関係を示す図である。

【図6】法線ベクトルの演算方法を説明する図である。

【図7】テクスチャデータの作成方法の説明図である。

【図8】テクスチャデータの例である。

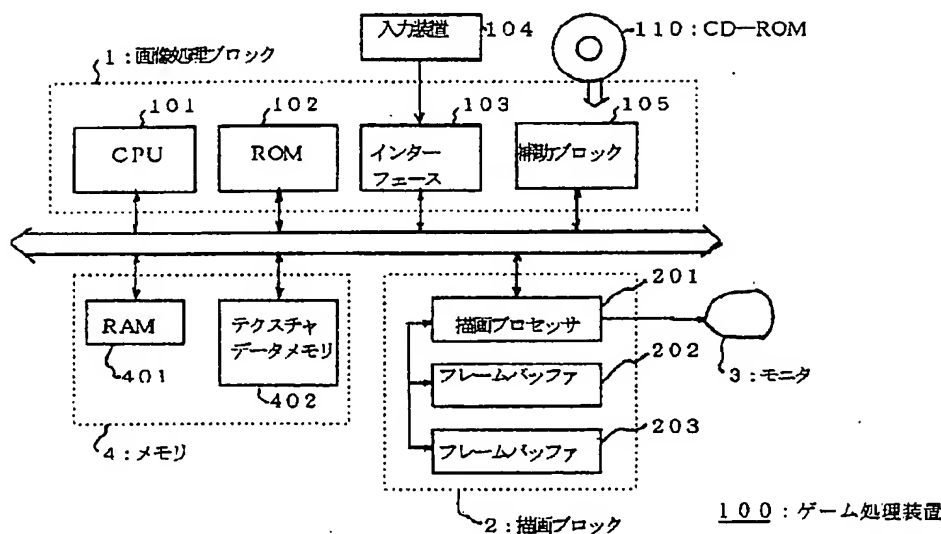
【図9】本形態の実施例の画像である。

【符号の説明】

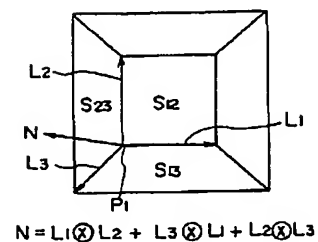
1…画像処理ブロック、2…描画ブロック、3…モニタ
装置、4…メモリ、10…ベクトル演算ブロック、11
…視線ベクトル演算部、12…法線ベクトル演算部、1

3…屈折ベクトル演算部、20…レンダリングブロッ
ク、21…交点特定部、22…テクスチャ決定部

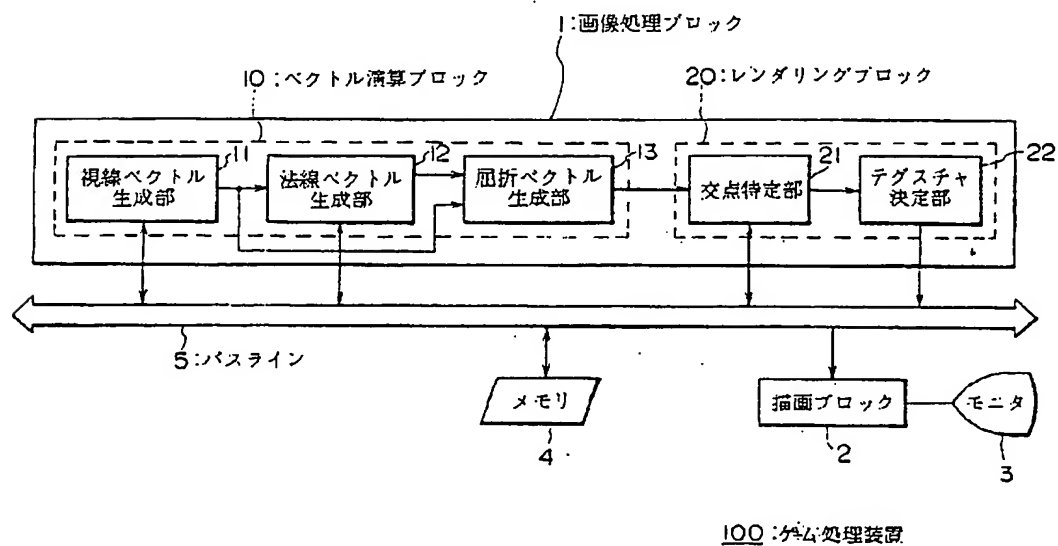
【図1】



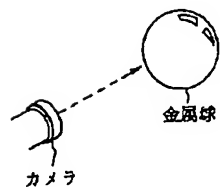
【図6】



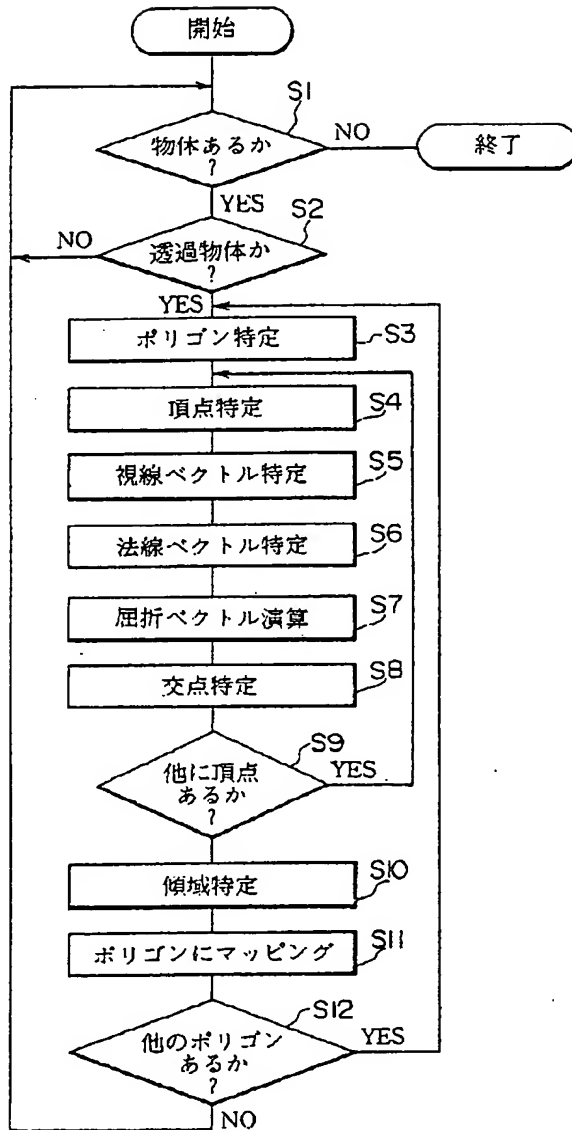
【図2】



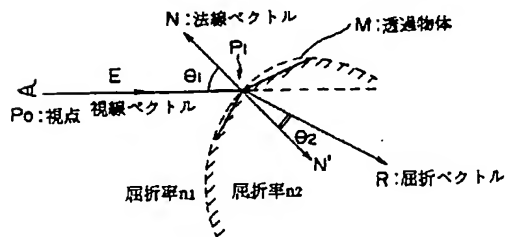
【図7】



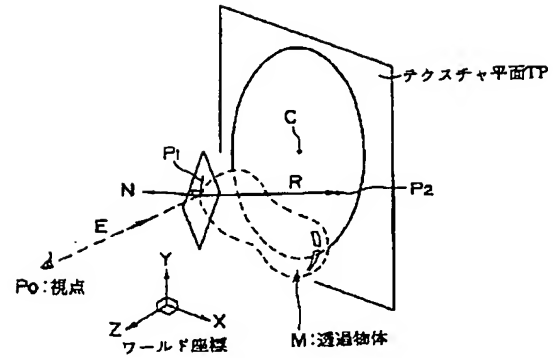
【図3】



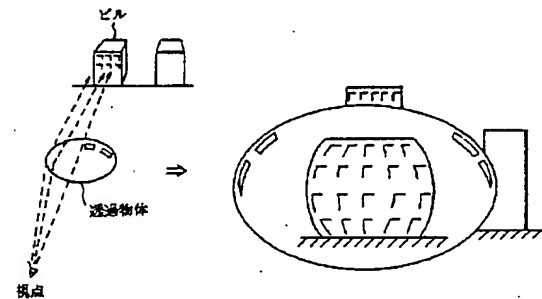
【図5】



【図4】



【図9】



【図8】

